Exercise 1:

M-File:

```
function y=average()
x=[];
for i=0:4
    G=input('Enter the grade ');
    x=[x,G];
end
disp(mean(x))
```

Running:

```
>> average()

Enter the grade 80

Enter the grade 89

Enter the grade 94

Enter the grade 77

Enter the grade 92

  86.4000
```

Exercise 2:

Code:

```
>> x=0;
for i=1:3
   for j=1:3
     x=i+j;
    if(x<4)
        disp('*')
    else if(x>4)
            disp('**')
        end
    end
end
end
```

Output:

```
*

*

*

**

**

**
```

Exercise 3:

M-File:

```
function y=matrix(L,M,N,K)
x=[L;M;N;K];
z=[L;L+M;L+M+N;L+M+N+K];
h=x.*(2+3i);
y=[sort(x),z,h];
```

Running:

```
>> w=3;
>> x=5;
>> y=9;
>> z=8;
>> M=matrix(w,x,y,z)

M =

   3.0000            3.0000            6.0000 +
9.0000i
   5.0000            8.0000           10.0000
+15.0000i
   8.0000           17.0000           18.0000
  27.0000i
```

Exercise 4:

-Dec2bin:
Convert a non-negative integer smaller than 2^52 to a binary string.
DEC2BIN(D,N) produces a binary representation with at least N bits.

Help:

```
help dec2bin
 DEC2BIN Convert decimal integer to
a binary string.
    DEC2BIN(D) returns the binary
representation of D as a string.
    D must be a non-negative integer
smaller than 2^52.

    DEC2BIN(D,N) produces a binary
representation with at least
    N bits.

    Example
       dec2bin(23) returns '10111'
```

Example:

```
>> dec2bin(8)

ans =

1000
>>
dec2bin(15,5)

ans =

01111
```

- **Bin2dec:**
  Convert binary string to decimal integer.

Help:

```
>> help bin2dec
 BIN2DEC Convert binary string to decimal integer.
    X = BIN2DEC(B) interprets the binary string B and 
returns in X the
    equivalent decimal number.

    If B is a character array, or a cell array of strings, 
each row is
    interpreted as a binary string.
    Embedded, significant spaces are removed. 
Leading spaces are converted to
    zeros.

    Example
        bin2dec('010111') returns 23
        bin2dec('010 111') also returns 23
```

Example:

```
>> 
bin2dec('1111')

ans =

    15
```

- **Dec2hex:**

  Convert a-non-negative integers smaller than 2^52 to hexadecimal string.

Help:

```
>> help dec2hex
 DEC2HEX Convert decimal integer to
hexadecimal string.
    DEC2HEX(D) returns a 2-D string array
where each row is the
    hexadecimal representation of each
decimal integer in D.
    D must contain non-negative integers
smaller than 2^52.

    DEC2HEX(D,N) produces a 2-D string array
where each
    row contains an N digit hexadecimal
number.

    Example
```

Example:

```
>>
dec2hex(10)

ans =

A
```

- **Hex2dec:**

  Convert hexadecimal string to decimal integer.

  Help:

```
>> help hex2dec
 HEX2DEC Convert hexadecimal string to
decimal integer.
    D = HEX2DEC(H) interprets the
hexadecimal string H and returns in D the
    equivalent decimal number.

    If H is a character array or cell array of
strings, each row is interpreted
    as a hexadecimal string.

    EXAMPLES:
                    and hex2dec('12b') both
```

Example:

```
>>
hex2dec('A')

ans =

    10
```

- **Factor:**

  Returns a row vector containing the prime factors of the parameter.

  Help:

```
>> help factor
 FACTOR Prime factors.
    FACTOR(N) returns a vector containing the prime
factors of N.

    This function uses the simple sieve approach. It
may require large
    memory allocation if the number given is too big.
Technically it is
    possible to improve this algorithm, allocating less
memory for most
    cases and resulting in a faster execution time.
However, it will still
    have problems in the worst case, so we choose to
impose an upper bound
    on the input number and error out for n > 2^32.

    Class support for input N:
       float: double, single
```

Example:

```
>> factor(25)

ans =

     5     5

>> factor(37)
```

```
>> help expand
 --- help for sweepset/expand ---

 EXPAND Expand records into sweeps

   S = EXPAND(S,t) turns each record into a sweep by
replicating it N
   times.  The number of times to replicate each record
is specified by the
   (1-by-NUMRECS) vector t.

    Overloaded methods:
       sym/expand
       dtree/expand
```

- **Expand**

  Expand

  Help:

Example:

```
>> syms x
>> expand(x+2)

ans =

x+2
```

Name: Haneen Ibrahim El-masry

ID: 220082480